6-8-2018

Communication System For Firefighters

Nicholas Goodpaster
Santa Clara University, ngoodpaster@scu.edu

Steven Booth
Santa Clara University, sbooth1@scu.edu

Griffin Moede
Santa Clara University, gmoede@scu.edu

John-Paul Hurley
Santa Clara University, jphurley@scu.edu

Santa Clara University
DEPARTMENT of COMPUTER ENGINEERING

Date: June 8, 2018

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Nicholas Goodpaster, Steven Booth, Griffin Moede, and John-Paul Hurley

ENTITLED

Communication System for Firefighters

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

THESIS ADVISOR

DEPARTMENT CHAIR

COMMUNICATION SYSTEM FOR FIREFIGHTERS

by

Nicholas Goodpaster, Steven Booth, Griffin Moede, and John-Paul Hurley

SENIOR DESIGN PROJECT REPORT

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California

June 8, 2018

Abstract

Currently firefighters use two-way radios to communicate on the job, and they are forced to write reports based on their memory because there is not an easy way to record the communications between two-way radios. Firefighters need a system to automatically document what happened while they were responding to a call. To save them a significant amount of time when creating reports, our solution is to implement an application that allows firefighters to take pictures, record video and communicate in real time with their team of on-site responders. The proposed system will use a Wireless Local Area Network (WLAN) hosted on the fire truck itself to act as an access point (AP) to which the firefighters can connect. This AP will also save communication between firefighters to a local storage location. Upon return to the fire station, the AP will route all of the information stored locally to a larger database. For now, Wi-Fi will be our communication medium, with a prediction that our technology can eventually be extended to include radio signal.

Acknowledgements

We would like to thank our project advisor, Dr. Silvia Figueira, for her support and advice throughout the course of our research, planning and development. We would also like to thank Sean Lanthier, a retired firefighter with the original idea for this project. Without his knowledge and push for innovation in the fire industry, we couldn't have produced a successful project. Lastly, we want to thank Allan Báez Morales, the Director of Programs and Partnerships for the Frugal Innovation Hub at Santa Clara University. Allan helped us get in contact with Sean, and provided necessary materials for us to succeed.

Table of Contents

List of Figures

Introduction

When firefighters respond to an emergency call, they rely heavily on walkie talkies to relay information to each other as they attempt to handle the situation. This technology provides a real time stream of communication, which is vital for first responders dealing with life or death situations. However, walkie talkies rely on public radio waves and firefighters are looking for a more reliable method for communication. Firefighters also waste a significant amount of time verbally reviewing the specifics of an incident and compiling a report upon their return to the fire station. With their current communication system, firefighters have no way of storing audio, photographs or videos captured during a service call. Hence, firefighters are looking for an updated system that will allow them to not only communicate with each other in real time but also document their trips to save them a significant amount of time when creating reports.

Our solution is to implement a web application interface that allows firefighters to capture pictures, record video and communicate in real time with their team of on-site responders. During a service call, all of the firefighters on-site will be able to use a smartphone or tablet to access the web application and communicate with the rest of their team. This communication will be real time and mimic the ease of use provided by their current walkie talkie system. All communication as well as captured images and videos will be compiled into a multimedia report that the firefighters will be able to review when they create their own response reports. This report will provide the firefighters with an organized, chronological list of communication, which will not only streamline the process of filling out reports but will also ensure that these reports are accurate.

The proposed system will utilize a Wireless Local Area Network (WLAN) situated on the fire truck itself to provide an access point through which the firefighters can communicate. Our web application will route all audio communication through this access point. The audio will be relayed to all of the firefighters within range of the system and will be saved on a local disk to be used for the report. Upon return to the fire station, the access point will link to the station's Wi-Fi network and all of the information stored locally on disk will be transferred to a larger database. This will allow firefighters with access to the Internet to view a recent history of reports without having to be directly connected to the WLAN. This solution will simplify the day-to-day operations of firefighters by providing a single application for managing their communication and digitizing their records.

In order to complete our proposed solution, there are certain requirements and objectives that need to be met. To start, our system has a few functional requirements which will help outline design and implementation. Some of these requirements are critical to the system's success, while others are recommended or suggested and will help improve certain aspects of the system such as ease of use. The critical functional requirements are as follows. The system will allow firefighters to communicate in real time with each other while responding to calls. This includes the ability to selectively communicate between different firefighters, instead of the current two-way radio system which only communicates with the entire channel. The system will also allow firefighters to take pictures and videos during a job. It will also record all conversations that occur during a

job. Lastly, the system will take all job media, including conversations, pictures, and videos, and save them to a repository that can be viewed later to help with report generation.

Following these critical functional requirements, the system also has a few recommended functional requirements. The lack of these recommended functionalities will not hinder the overall functionality of the system, but their presence will greatly aid the client in their use of the system. The first recommended requirement is that the system will color code firefighter emblems with their assigned duty during that job. This would make it easier for the lead firefighter to selectively communicate among their peers on the fly. Another recommended requirement is that the system will save account information for each firefighter that includes their name, position, photo, and skills. Lastly, the system has a suggested requirement as well, which is less important than the recommended requirements while also not necessary. This suggested requirement is that the system will translate audio communication into text format for reporting.

The system also has non-functional requirements which do not affect the functionality of the system but will help determine how the functional requirements are achieved. The non-functional requirements our system will meet are as follows. The system will be quick and efficient, to ensure instant communication. The system will be intuitive so that minimal training is required for use. And lastly the system will be reliable so that firefighters can trust it will work in critical situations.

Finally, the system must also follow a specific design constraint. This constraint will limit the ways in which the system will be able to operate. The main design constraint is that there will be no Internet access, because the centralized Wi-Fi router will not be connected to the Internet. By keeping these requirements and design constraints in mind, we are able to continue on with the planning of our design, confident we will not run into any major issues deep into development.

Main Body

Real-Time Selective Communication

In order to meet our objective of allowing firefighters to communicate in real-time with others on their job, there were a few tasks that needed to be completed. The first was creating a wireless local area network that each client would be able to connect to without relying on cell service or internet connection. Second, we set up a server that used signaling to route messages from one client to another. Lastly, we made use of an API called WebRTC [3] to set up an audio connection between two peers.

Wireless Local Area Network

When determining what technology we wanted to use to connect clients, we looked into many options. Though radio waves were the most efficient in terms of range and reliability, as developers, we don't yet have access to manipulate the radio capabilities of the users' devices. So instead we decided to use WiFi, which has similar capabilities and is a more convenient option. So in order to use WiFi, we needed to set up a wireless local area network (WLAN) for our clients to connect to. The first step in setting this up was to choose a device that could emit its own network. We decided to use a Raspberry Pi [7] because of its portability (the device is 3.370 in × 2.224 in × 0.669 in), its ability to run the servers we needed for our application, and as mentioned for its ability to host a small WLAN. Because of this, each device simply needs to connect to the WiFi network emitted by the Raspberry Pi and it would have access to any web servers running on the device. For future development, we also plan to connect a router to the Raspberry Pi in order to strengthen its signal, but for our testing and early development, the Raspberry Pi's network capabilities were sufficient.

Node Server with SocketIO

Now that our network was set up and each client connected to it, we needed a way for them to communicate. First, we set up a NodeJS [6] server with the ExpressJS [5] framework in order to allow our clients to access the web application. This server then also allowed us to run our SocketIO [8] code. SocketIO is a platform that enables event-based communication between clients through a server. One client sends a message to the server, and the server relays that message to the desired receiving client. This feature was the base that enabled selective communication. SocketIO allows a client to choose who they want to connect with. When a client connected to the WLAN we set up accesses the server on their device, the SocketIO code adds them to what is known as a room. All clients that are connected to the server are placed in this room, and to communicate with other clients, they can do so by sending messages to the whole room or to specific clients within the room.

WebRTC

With SocketIO running on our NodeJS server, the only task left was to set up an audio connection between two clients. To do so, we used an API called WebRTC. WebRTC works by sending messages back and forth between two clients that include information about each client's device, audio stream, socket connection and more. Once each client has enough information about the other, the connection is set up and each client can begin sending data to the other. For the purposes of our project, the data we are referring to is an audio stream of what each client is saying (like in a phone call). In order to get this media, we simply had to run a JavaScript function called getUserMedia, passing in the type of media we want (audio). This function returned an audio stream which was then sent across the connection to the other client. Now with audio being passed over a secure connection to another chosen client, we completed our objective and requirement of allowing for real-time selective communication.

Recording Conversations

Another objective we wanted to meet was to be able to record all conversations that occured during a job so the firefighters could go back later and listen to details of the job. In order to do this we used an API called MediaRecorder [4]. This API allowed us to record the audio on a device that was already being gathered to send to the other peer in the conversation. We gathered chunks of audio and combined them into a JavaScript blob (chunk of bytes that holds data). One blob of audio data was created for each client's local audio. We saved these blobs as mp4 audio files and saved them to a folder on the Raspberry Pi titled with the date and time that the job began. These folders are located within a parent folder called 'conversations'. Then, later when the conversations were needed for report generation, we take the two audio files that were a part of each conversation and mold them together to create one conversation file. These conversations are available to listen to in the previous jobs portal that we will discuss in the Media Persistence section below.

Visual Media

The third objective we needed to meet was to allow for firefighters to capture visual media (pictures and videos) while on a job. This objective required two steps. First, we created a new HTML page to handle all visual media. This page had a view for capturing visual media and a view for viewing visual media that had already been captured on the job. Second we created local folders on the Raspberry Pi in order to save data. A folder was created right when the job began (when the server was started) and was titled with the date and time and placed within a parent folder titled 'media'. When a client wants to take a picture or video, we again use the JavaScript function getUserMedia, only this time passing in 'video' as the type, which gives the user access to the camera. When an picture or video is taken, it is added to the job folder mentioned above.

When the client wants to look at all visual media taken on the job, they simply close the camera and all visual media from the folder is loaded into a carousel and gallery for them to scroll through.

Media Persistence

Another objective was for the media taken during the job to be readily available at a later time in order to make report generation more efficient and accurate. With the images saved to the Raspberry Pi, the only other step taken to complete this objective was to provide a separate interface that would allow firefighters to select a previous job and view all media associated with that job. In order to complete this, we created a separate HTML page for previous jobs. When a client wants to look at previous job media, they open this page, and select a job from a drop down menu. When the job is selected, the photos, videos, and conversations that were gathered from that job are loaded from their respective folders and populated into three tabs (pictures, videos, and conversations). For our testing and early development purposes, this data is only stored locally on the Raspberry Pi, so in order to view this job data, one must till be connect to the Raspberry Pi's WLAN. However, in the future, when we scale up ir development, we plan to use a large external database to hold all of these files and host this section of the application with an external web server, so that the firefighters can access the data whenever they have access to the internet. When the firefighters return from their job, the media will be transferred from the Raspberry Pi into this database, and then wiped from the Raspberry Pi so as to save space on the device.

Accounts

Finally, the last requirement was to allow for account creation so each firefighter had their name, position, photo, and skills saved and uploaded for each session. This objective required two steps - first we set up a local MongoDB [2] database to hold all of the account information, and then we created HTML pages for creating accounts and logging into the system.

MongoDB

In order to save account information for each firefighter, we first set up a MongoDB database. MongoDB is a database platform that holds data in javascript objects called documents. These documents are grouped into collections. For our purpose, we created a collection called personnel, containing one document per firefighter. The document held information like username, encrypted password, name, position, and skills. The database was set up and run locally on the Raspberry Pi, making it easy to access the data when verifying firefighters.

Account Creation and Login

To make use of the database, we created an HTML page for creating accounts and logging into the application. The account creation view was very simple and asked for all the data fields mentioned in the MongoDB section. The fields were then taken and formed into a document and inserted into the database. Now when a firefighter wants to use the application, they open the

login view of this new page, and enter their username and password. If the user is found in the database and the password matches, they are logged into the system and taken to a page asking them if they want to view the personnel page or the visual media page. This account creation feature is also vital in allowing for real-time selective communication. Each user has their own view of the application that shows the pictures and names of all their fellow firefighters.

Model of Technologies Used

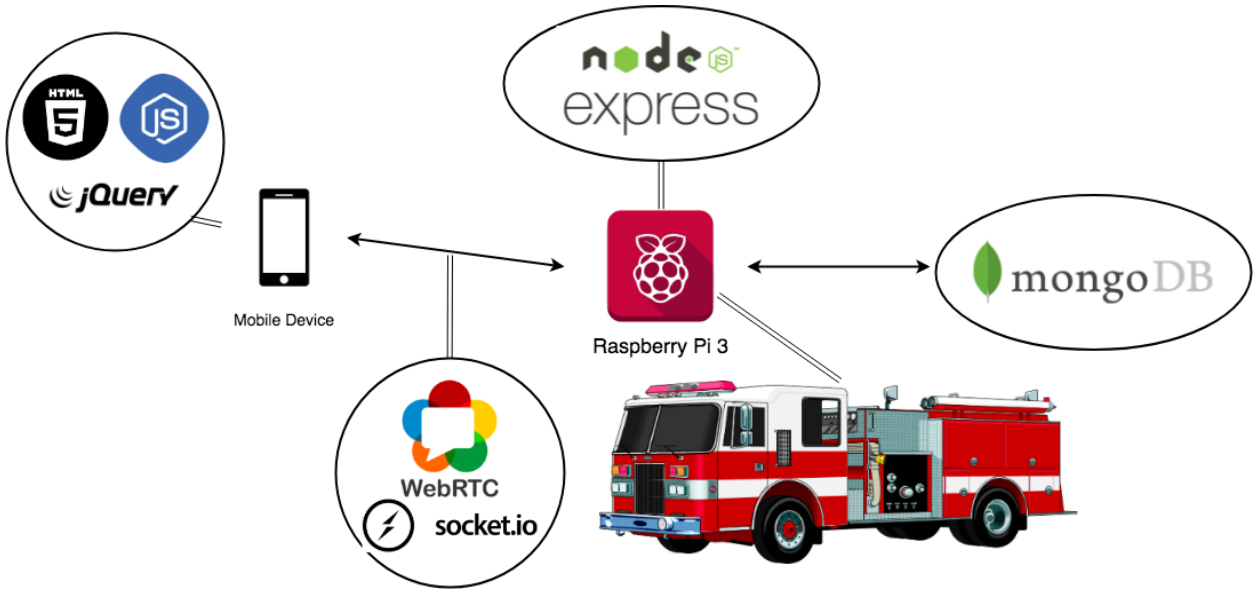

Figure 1 - Technologies Used Diagram

Design Rationale

General Rationale

During our preliminary research for this project, we investigated different forms of local communication that could be suitable for firefighters. It was determined quickly that Wi-Fi and bluetooth both lacked signal strength in long or medium ranges, and that cellular service was too unreliable and slow to realistically use. The two way radios which are currently used by firefighters are powerful and reliable compared to the other forms of communication.

Two way radios do have flaws: they are inconsistent and somewhat unreliable, especially when being used in large buildings or when there are thick physical barriers like concrete between radios. Two way radios operate on a channel system, where everyone is on the same channel, and communication typically consists of an operator being able to either speak, or listen, but not both. This is known as half-duplex mode. Full duplex mode, where you can both speak and listen, is available but not typically used by firefighters. We wanted to ensure that firefighters could both speak and listen, and selectively choose multiple different people to talk to. We also wanted the application to record the communication (not currently supported with two way radios), and take pictures and videos.

The issue we faced was that two way radios are currently the most realistic communication medium for firefighters, but there isn't much room for improvements because they are not built in to smartphones. The features we wanted could be achieved with Wi-Fi, but the signal is not ideal for actual communication because it isn't typically strong enough. We predict that in the future, smartphones will be equipped with two way radio technology, which would allow us to try and integrate our features with the communication medium. For now, Wi-Fi will be our communication medium, with a prediction that our technology can eventually be extended to include radio signal.

Architecture Rationale

With Wi-Fi as our communication medium, we decided to use a WLAN in infrastructure mode, with a central router/server combination mounted as a hub on the fire engine. Firefighters would then communicate with each other through this router. This is essentially a client-server architecture, with the clients being the firefighters and the server being the router/server combination.

Technologies Used Rationale

The MEAN stack provides the scalable database MongoDB, along with a combination of web frameworks including Express, AngularJS, and NodeJS. Express is a node.js web application framework including features for building single, multi-page, and hybrid web applications. AngularJS extends HTML vocabulary for our application. This leads to a more expressive and readable environment. NodeJS is for building fast and scalable network applications.

The goal of this project is to create a hybrid web application that will allow for easier use of built in device hardware like the camera, while still allowing the application to be web based using HTML, CSS and Javascript. We have no experience using native android development technologies, and we do have experience with the classic web development languages.

We chose python for the networking programming for its simplicity and efficiency in implementation in comparison with a low-level language like C.

Speex will be used to lessen the size of audio files, and highlight voices in the audio. It is a free audio compression codec.

Conceptual Model

Since the web application will be used by first responders on scene, simplicity in the systems design and user interactivity were key design considerations for the conceptual models. By ensuring our interface is easy to use and navigate, the likelihood of our project's eventual adoption will increase significantly.

Login and Create Account Page

The first page that users will be directed to is the Login and Create Account page. If the firefighter does not yet have an account, they select the tab labeled Register (Figure 3.1) and enter their information. Once they complete this step, their account is created. Now that they have an account, they choose the tab labeled Login (Figure 3.2) and enter their credentials. It is at this login step where they can choose to either enter the current job or visit the previous jobs portal by selecting the radio button of their choice. This interface is meant to be highly intuitive and allow the first responders to navigate in an efficient manner.

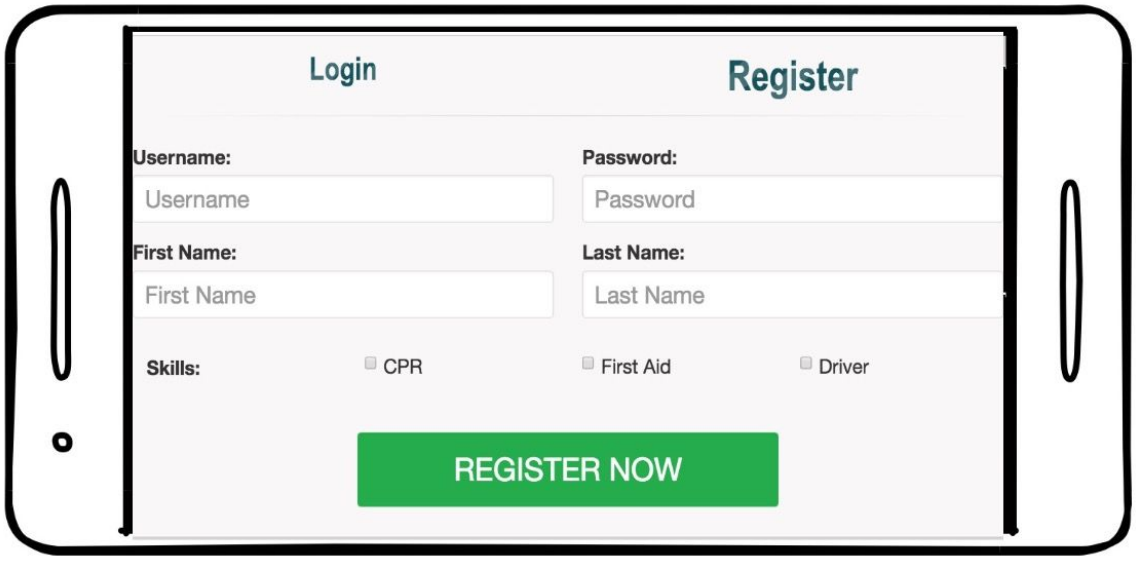

Figure 2 - Create Account Page

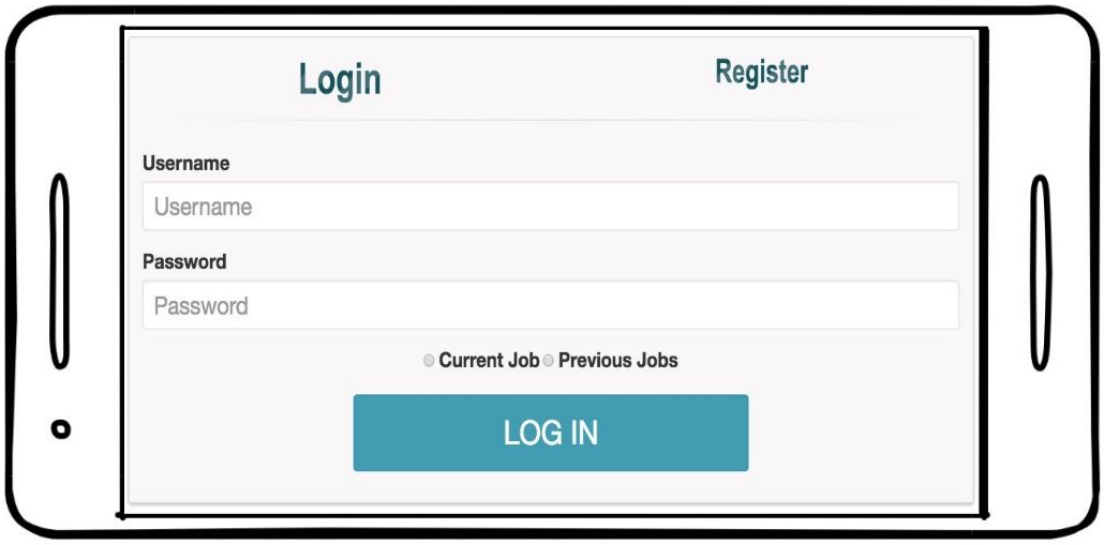

Figure 3 - Login Page

Active Job Selection Page

If "Current Job" is selected from the Login page, the firefighter will be routed to the navigation page for a job in progress (Figure 3.3). This page provides the firefighter with the option of either capturing or viewing by navigating to "MEDIA" or communicating and viewing the status of their team by selecting "PERSONNEL."

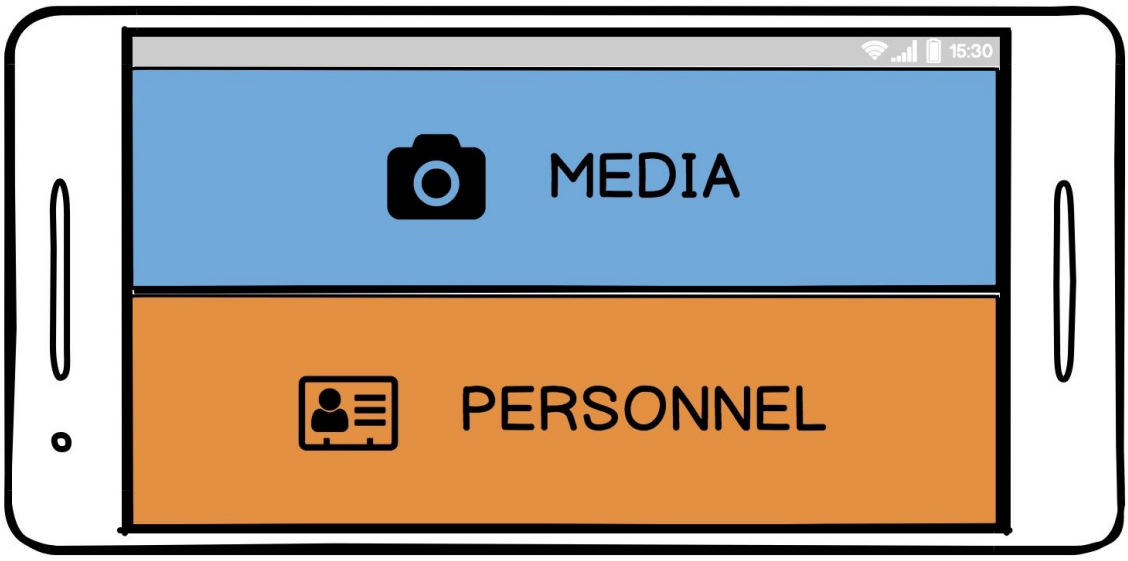

Figure 4 - Active Job Selection Page

Visual Media Page

The Visual Media Page, selected by pressing “MEDIA” on the Active Job Selection Page, features a visual display of data accumulated on the current call as well as a large button giving the firefighter the ability to capture pictures and videos from the scene (Figure 3.4). Once the camera icon is pressed, the user will be routed to a screen with the phones native photo capture scene. This page aims to allow the firefighters to quickly view and snap photos and video for a call in progress.

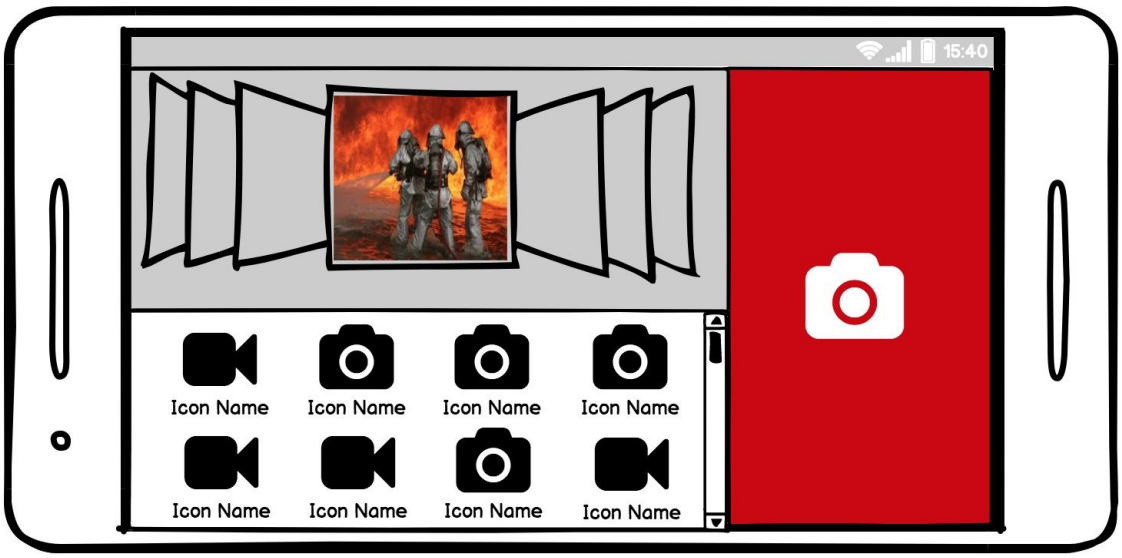

Figure 5 - Visual Media Page

Personnel Page

The Personnel Page features a dashboard with all of the active crew members with their respective color based on their role in a current call (Figure 3.5). Upon the selection of a crew member, a side bar will appear giving the firefighter basic information about the crew member as well as the option of initializing communication with the selected crew members. Each crew member will have a photo identification associated with their button to make the system even more intuitive and easy to use.

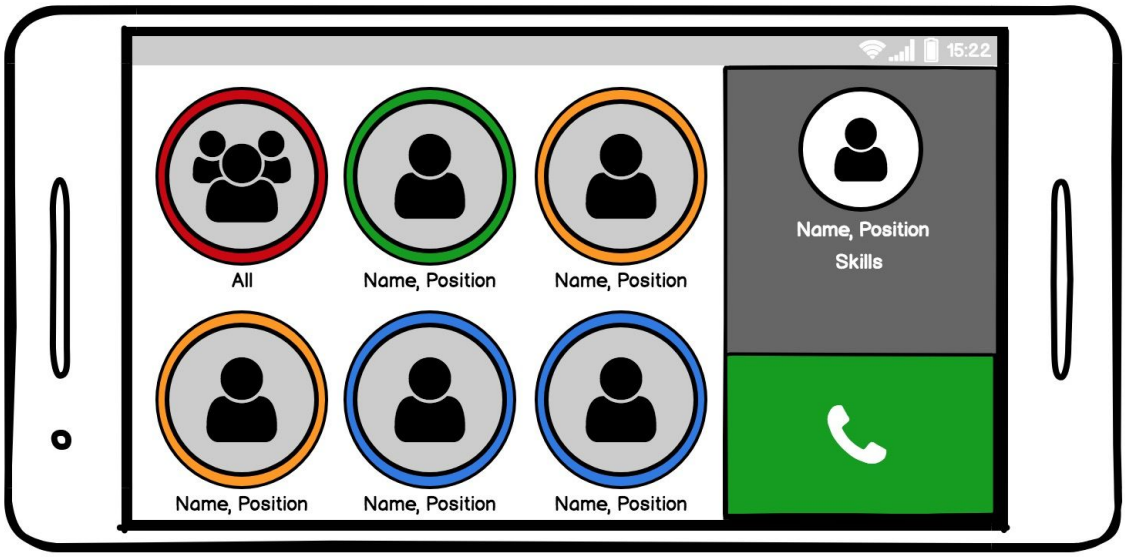

Figure 6 - Personnel Page

Previous Jobs Portal

The final page of the system is the “Previous Jobs Portal,” which can be accessed by selecting the “Previous Jobs” radio button from the Login Page. This page will be optimized for desktop to allow firefighters to view information about previous jobs from anywhere with access to the internet. In this portal, firefighters will be able to search previous jobs by date. Within each case there will be a chronological list view of all the media and communication logs captured on that particular call (see Figure 3.6).

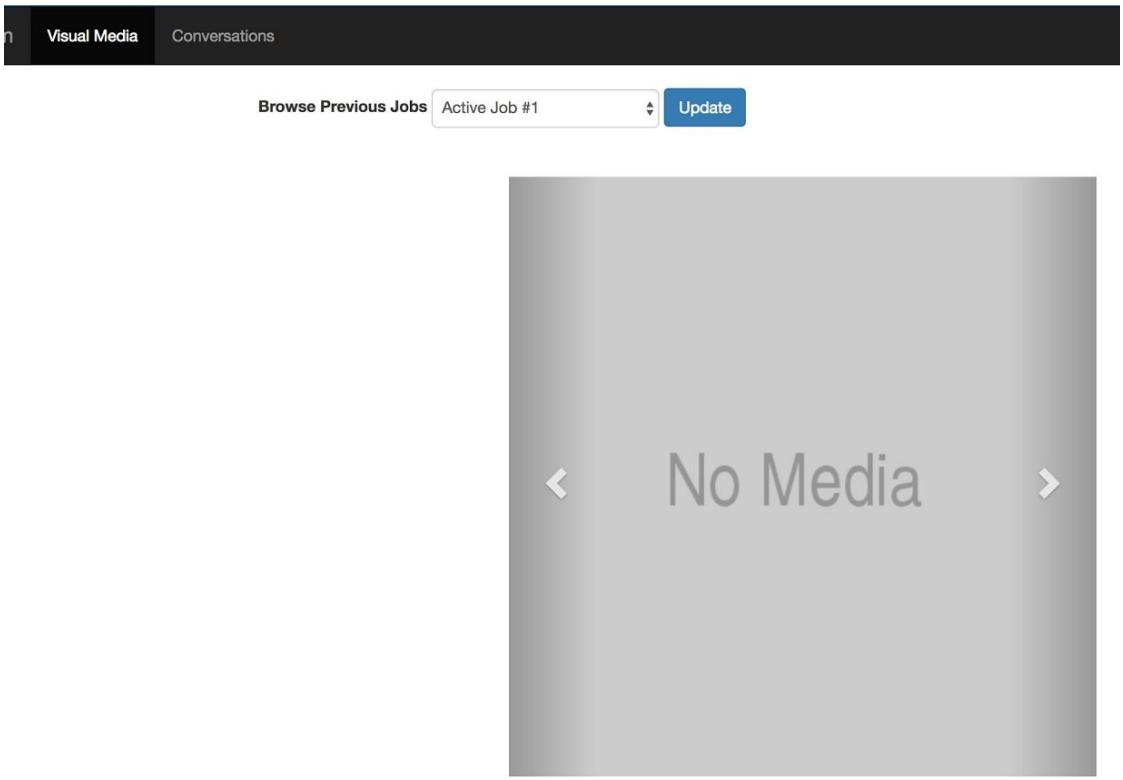

Figure 7 - Previous Jobs Portal

Societal Issues

Ethical

Assisting firefighters in communication would have a direct impact on the lives of firefighters and the people they are assisting, whether that is rescuing someone from a burning building or protecting homes from wildfires. This impact could be positive or negative, depending on whether the application works as it should. Because of this potential risk, there is a large burden placed on us as developers to make sure that the technology is reliable and has plenty of fail-safes built in. Even though our project was more of a proof of concept than something to be deployed in the field, we tried to model our technology around this concept of reliability, and failing gracefully if necessary.

Social

This project doesn't just affect firefighters, it affects everyone who could potentially rely on firefighters as well. This encompasses pretty much everybody in developed areas. Allowing firefighters to communicate more efficiently or accurately with each other could help them in a broad range of ways. Report generation and the ability to rewind through communication, photos, and videos could also help with the accuracy of reports. It is understandably difficult to generate a report based on memory at the end of the day after potentially multiple jobs. These reports can be used to help solve crimes, and pictures and videos captured on the scene could help paint a better picture for investigators.

Political

Politically, our project has an interesting role. Everyone would clearly want firefighters to be able to do their job more accurately and efficiently, but not everyone is ok with technology taking a more of a prominent role in firefighting. It was brought to our attention that firefighters may not want to have all of their conversations recorded while they are on the job, as they may sometimes use methods that are not part of protocol- but could be better suited for solving the problem at hand. There is a similar issue emerging with police, whether or not officers would consent to having body-cameras recording at all times. Also, this application would be replacing a 90-year-old technology whose limits and shortcomings are well understood. Our application on the other hand is not as thoroughly tested and could potentially cause problems.

Economic

The cost of this system for firefighters wouldn't be unreasonable, as the only potential costs would be a Raspberry Pi and basic smartphones. The only costs for the project was the hardware, and there was almost no cost for data storage, because the data is kept on the firefighter's local computer. Our solution is an affordable alternative to the current technology used by firefighters. It is designed to utilize common and readily available technology, basic smartphones and a Raspberry Pi, instead of being a standalone device which could cost more.

Health & Safety

There aren't currently laws which govern how reliable a technology assisting first responders should be. This begs the question of whether we would be responsible for a technological failure that could result in loss of life. Situations like these are usually prosecuted on a case to case basis, to determine whether we were at fault. Typically when a software fails to do what it's supposed to do, it is considered a breach of contract. This then enters the jurisdiction of contract law, which has been successfully employed by plaintiffs to sue software vendors who deliver products that didn't work correctly. But this isn't necessarily the path that the law would take- it is particularly unclear because laws haven't caught up with technology.
It would obviously be of the utmost importance that our product functions correctly and reliably, but there is no guarantee. We were lucky that our project isn't anywhere near ready to be deployed, so we didn't have to enforce strict reliability- we simply aimed to get the product working as a proof of concept. Using WiFi as a communication medium for first responders wouldn't be realistic, as the signal strength

Manufacturability

The product only needs to be assembled and downloaded, as it is already fully built. We were expecting early on to have to use a router along with the Raspberry Pi as a means of creating a wireless access point, but in the end we were able to use the Raspberry Pi alone. This simplified the assembly process of the system. Building the application in the first place took a longer amount of time than expected, but because it is just software it only needs to be built once. The cost was very reasonable for the Raspberry Pi ($60), and the application can be downloaded onto any smartphone.

Sustainability

In a narrow sense, our product is still in a flexible stage of development and could be configured to use a different method of communication instead of WiFi. If developers eventually are granted access to the radio communication in smartphones, our product could be modified to

use that instead of WiFi for an increased signal strength. Our project was designed to be integrated with future developments, because there currently isn't a communication signal strong enough to implement the system for firefighters. We are using WiFi as a temporary stand-in. If our product was implemented, we would increase the safety and effectiveness of fire and rescue missions. Lives and property could be saved, sustaining the community that the firefighters support along with the firefighters themselves. This could also make it safer to be a firefighter, potentially attracting people to the profession knowing that it is slightly safer- and keeping current firefighters alive. Firefighters would also have to spend less time filing reports, as all of the communication is recorded and accurate.

Environmental Impact

Our environmental impact is minimal, as most people already have smartphones that can be used with the application. Fire departments would theoretically need a Raspberry Pi for each engine at the department, but Raspberry Pi's are small and use minimal materials. There are multiple initiatives and regulatory constraints that are placed on electronics manufacturers to ensure that there is minimal waste that ends up in landfill [1], and the PCB assembly is ROHS-compliant, which forbids the use of lead or harmful metallic elements in PCB manufacturing [10] . Sony, the company which manufactures Raspberry Pi's, also pledges to avoid contributing to conflicts or serious human rights abuse through sourcing practices [9].

Usability

First responders are tasked with making serious decisions when lives are on the line. The structure of our system is based around usability and ensuring that the system is as easy to understand in a critical situation as possible. There several desired improvements that can be made to the system after some acceptance testing with our client (See Future Work). For now, this project can be seen as an introduction to how the smartphone can be utilized to provide first responders with more tools with which they can better communicate and compile reports while on the job. By identifying a key difficulty faced by many first responders, the need to compile reports by memory, this project has provided a user friendly solution that will enable first responders to do their job faster and smarter.

Lifelong Learning

Since this project is on the cutting edge of first responder technology, it shows the importance of lifelong learning in the Computer Science industry. It is clear that there are fields of work that have yet to be impacted by advancements in technology. In this case first responders and the two-way radio have been working in tandem for decades without any adjustment to new technologies. This brings about the question of which other industries could benefit from a technological revamping. In addition, many of the technologies used to accomplish the functionality of this project are very modern and widely documented cross platform technologies.

None of the technologies were a part of the Santa Clara University curriculum, and in order to gain experience we had to do outside research and come to an understanding of the software on our own time and out of our own self-drive. As Software Engineers, it is our duty to stay driven towards learning new technologies and staying aware of industries that may benefit greatly from new technologies.

Compassion

First responders are tasked with the responsibility of ensuring that people's lives are safe in emergency situations. Throughout our design process, it was essential to keep the importance of this work in mind to ensure that our system was suitable for such critical situations. We took this project upon ourselves not only because we believed that the technology was cutting edge and relevant to our careers as software engineers, but we also felt the importance of giving back to first responders who do so much for so many people on a day to day basis and often lack the recognition for doing so. Through this project we hope to have provided first responders with at least a proof of concept for the capability of mobile phone to transform the first responder industry.

Conclusions

What Was Accomplished

Through months of research, planning, development, and testing, we created a real-time communication system that allows firefighters to engage in instant and selective calling, record conversation, capture photos and videos during a job, and review all media gathered on a job at a later point in time. We ran a NodeJS server on our Raspberry Pi with ExpressJS to host our web pages. The We used the SocketIO and WebRTC JavaScript APIs in order to connect clients and pass audio data over secure connections across our server. We also made use of the camera on clients devices and the filesystem on the Raspberry Pi to save visual media for later report generation. Lastly we saved information about firefighters to allow for account verification using MongoDB database. Each of these features completed requirements that helped us develop a product that met nearly all of our clients needs and desires. After it was all said and done, our final project used over 3,600 lines of code and numerous different API's and other technologies to help accomplish our goal.

What We Learned

Throughout our development process, there were many obstacles encountered along the way. As aspiring software engineers, we have taken many valuable lessons away from this project and throughout our design process.

Hardware and Software Don't Always Mesh

Our project involves both software and hardware components. Development began on the software side of the project as we began to narrow down the technologies that were necessary to complete our overall vision for the project. The project was successfully working on a personal computer before we made the transition onto the Raspberry Pi. Upon transitioning our hardware, it was evident that the cross compatibility that we had hoped for was far from reality.

Most of the technologies that we used and had working on our personal computer were able to download and run successfully on the Raspberry Pi's native operating system, which ran as a 32-bit ARM-V7. We soon found out that our most recent version of MongoDB, our chosen database solution was not compatible with a 32-bit operating system. To mitigate this issue, we first tried to download a new, 64-bit operating system onto the Raspberry Pi. This led to even more issues as the Raspberry Pi's hardware struggled to handle this new operating system.

Our final solution was to roll back the operating system to the native, 32-bit version and we were able to find an older version of MongoDB that was compatible atop a 32-bit operating system. This troubleshooting showed us first hand the importance of planning and research when integrating hardware and software solutions.

Tutorials Should Be Taken with A Grain of Salt

Much of our project development revolved around online tutorials to get a better understanding of the new technologies that we planned to use for the project. Though these tutorials were beneficial in giving us an idea of how we could best utilize the technologies, some of the tutorials were misleading and often led us astray from our intended goals.

One example of this led to what could have become a major pitfall of our project: being able to run MongoDB on the Raspberry Pi. When going through the installation of MongoDB on our personal computer, a tutorial advised us to download the latest version of the software. Without hesitation, we proceeded to download the latest version and began integrating this technology into our project. It wasn't until we approached the final stages of hardware integration that we realized this minor step would lead to serious setbacks in the project.

By following the tutorial at face value, we were misled into thinking that this version of MongoDB would be compatible across hardware platforms. Little did we know that the latest version of MongoDB would not run as seamlessly on the Raspberry Pi and this led to complications. All in all, this experience has taught us the importance of taking tutorials with a grain of salt and doing extra research before taking tutorials at face value.

Proper Documentation is Imperative

With a project that involves software and hardware components that each have a high degree of specificity, it is absolutely imperative to keep proper and up to date documentation. When assigning tasks for the project, we split up the implementation in half, with half of the group members focusing on hardware and the other half on the software components. Once we began to make headway on the software end of the implementation, it became clear that all of the different moving parts to the software were going to be very difficult to communicate to those working on the hardware portion of the project. This realization led us to ramp up our documentation to ensure that the integration between the two parts of the project would go as smoothly as possible.

At the end of the day, it is unrealistic to keep everyone on the team on the same page through word of mouth alone, and in order to truly maintain a cohesive development process, focus had to be placed on documentation. Our documentation and version control were stored on Github and it became each individual group members' responsibility to stay on top of not only their own portions of the project but also to ensure that they were up to date with the contributions of the rest of the members.

Advantages and Disadvantages

Advantages

Our communication system for firefighters offers many improvements to firefighters' current forms of communication and report generation. For starters, the communication aspect of our product allows firefighters to choose who they want to talk to. Currently, walkie talkies only

allow for broadcast communication to anyone on the same channel as the person speaking. By allowing firefighters to decide who hears their messages, those who are not involved are not distracted by messages not meant for them. Secondly, the media portion of our project offers new functionality meant to improve report generation. Firefighters can now take pictures and videos of the job and can view them at any time. When they're on the job, this gives them the ability to see what else is going on at a different area of the jobsite. When they are finished with the job, these photos and videos give detailed evidence of what the site looked like, and what occurred on the job, helping the firefighters to create more accurate reports.

Disadvantages

Though our system provides great improvement in many areas, a few disadvantages do exist. The first of which is the fact that on some jobs, especially those requiring quick and critical response, the act of taking out ones phone to take a picture or video may not be realistic. If the first responders don't have enough time to take pictures and videos, that portion of the product will go unused and it won't live up to its full potential. Another, more pressing disadvantage, is that though Wi-Fi is a quality alternative to radio waves, its range and reliability are not perfect. If firefighters are responding to a critical situation, they may need to rely on their communication network being available at all times. If the Wi-Fi connection went out of range or malfunctioned for some reason, the consequences could be devastating. The last disadvantage we see is the way we are currently storing photos, videos, and conversations. For early development purposes, we are only storing this media locally on the Raspberry Pi itself. This means that this media can only be viewed when one is connected to the device's Wi-Fi network. Most report generation occurs after the jobs are complete and the firefighters are back at the station, and being connected to the Raspberry Pi at that time is not ideal. Eventually, as we move closer to production, the idea is we would store all previous job data in an external database and run the previous jobs portal on an external web server that can be accessed at all time from any internet connection. This would give firefighters the full-time access to this data that they need for efficient report generation.

Future Work

Polish UI

We plan to spend some time polishing the User Interface of the system to make sure that it is most suitable for being a usable system for firefighters. Improvements may include changing the color scheme or adjusting the sizing of certain elements to improve ease of use for the product.

Testing

The project still needs to go through some acceptance testing with our client to ensure that the project requirements were adequately satisfied. To get this validation, we plan to meet with

firefighters and have them try out our system in a closed environment and collect feedback on the usability of the system.

Color Code Firefighters by Position

It is typical for firefighters to be color coded by their position on a specific job. This color coding is often handled differently department to department. We have all the tools necessary to implement this color coding on our Personnel page but lack the domain specific knowledge to assign colors to team members by position.

Account Specifics

When registering for an account on our system with the current implementation, firefighters are asked to select skills upon creating their account. These skills are attributed to their account and other firefighters on the team are capable of viewing these skills during an active job. Currently these skills are not domain specific and we are unsure if they are relevant enough to firefighters during an active job. We hope to gain more domain specific knowledge during acceptance testing and to be able to fill these account specifics with the most relevant information possible.

Convert to a Native Android Application

Currently our web application runs off of a browser on the mobile phone. We hope to use a third-party API called PhoneGap to convert our web application into a native Android application. This will help with usability, since firefighters will no longer be required to open their web browser and navigate to the proper URL. Instead, they will simply be able to access the web application from their native Android interface, which will significantly streamline the process.

Translate Audio to Text

One of our suggested requirements was to be able to translate the saved audio conversations into text format for reporting purposes. We currently have all of the raw audio data to complete this step, and all we need to do to satisfy this requirement is to find an API that takes care of converting an audio .wav file into a text file. Once this is completed, the next step is simply to display this text transcription alongside the audio.

References

[1] European Commission. *Waste Electrical & Electronic Equipment (WEEE).* http://ec.europa.eu/environment/waste/weee/index_en.htm.

[2] MongoDB. *MongoDB Documentation.* https://docs.mongodb.com/

[3] Mozilla. *MDN Web Docs - WebRTC API Documentation.* https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API.

[4] Mozilla. *MDN Web Docs - MediaRecorder API Documentation.* https://developer.mozilla.org/en-US/docs/Web/API/MediaRecorder

[5] NodeJS Foundation. *ExpressJS Docs.* https://expressjs.com/en/api.html.

[6] NodeJS Foundation. *NodeJS Docs.* https://nodejs.org/en/docs/.

[7] Raspberry Pi Foundation. *Raspberry Pi Documentation.* https://www.raspberrypi.org/documentation/.

[8] SocketIO. *SocketIO Docs.* https://socket.io/docs/.

[9] Sony. *Responsible Sourcing of Raw Materials.* https://www.sony.net/SonyInfo/csr_report/sourcing/materials/.

[10] UK Department for Business, Energy & Industrial Strategy, Office for Product Safety and Standards, and Department for Environment, Food & Rural Affairs. *Regulations: Restriction of Hazardous Substances (RoHS).* https://www.gov.uk/guidance/rohs-compliance-and-guidance